Econometric Feedback for Runtime Risk Management in VoIP Architectures

Oussema Dabbebi, Badonnel Rémi, Festor Olivier

Econometric Feedback for Runtime Risk Management in VoIP Architectures

Oussema Dabbebi, Rémi Badonnel and Olivier Festor

INRIA, Nancy University

Abstract. VoIP infrastructures are exposed to a large variety of security attacks, but the deployment of security safeguards may deteriorate their performance. Risk management provides new perspectives for addressing this issue. Risk models permit to reduce these attacks while maintaining the quality of such a critical service. These models often suffer from their complexity due to the high number of parameters to be configured. We therefore propose in this paper a self-configuration strategy for supporting runtime risk management in VoIP architectures. This strategy aims at automatically adapting these parameters based on an econometric feedback mechanism. We mathematically describe this self-configuration strategy, show how it can be integrated into our runtime risk model. We then evaluate its deployment based on a proof-of-concept prototype, and quantify its performance through an extensive set of simulation results.

1 Introduction

Voice over IP (VoIP) has become a new paradigm in the area of telephony. It contributes to the convergence of services and permits IP providers to offer telephony service with a higher flexibility than traditional PSTN (Packet Switch Telephony Networks). The interoperability amongst VoIP equipments is supported by the standardization of dedicated protocols, including signaling protocols such as SIP (Session Initiation Protocol) for establishing phone sessions, and media transport protocols such as RTP (Real-time Transport Protocol) for transporting communications. The large scale deployment of VoIP infrastructures has introduced new security issues, including security threats inherited from the IP layer, such as denial of service and IP spoofing, but also security threats specific to VoIP protocols such as SIP flooding and SPIT (Spam over Internet Telephony) [**?**,**?**]. A large variety of security mechanisms (firewalls and intrusion prevention systems) are available for preventing these threats. However, the application of these security mechanisms on such critical infrastructures may seriously impact on the performance and usability of telephony service.

Risk management provides new opportunities for addressing this trade-off between security and performance [**?**]. It aims at quantifying the potentiality of threats and selecting suitable security safeguards in order to minimize the impact the VoIP infrastructure. In that context, we have already argued in favor of the application of risk management at runtime, by extending and applying the Rheostat model to VoIP architectures [**?**]. The parameterization of a risk

model is a key challenge.We therefore propose in this paper a self-configuration strategy for supporting runtime risk management in VoIP infrastructures. This approach permits to simplify the configuration of such risk models, by refining at runtime the model parameters based on an econometric feedback mechanism. We define this strategy in a theoretical manner and then describe how it can be integrated into our runtime risk model for VoIP architectures. We consider the model parameter which characterizes the cost of a safeguard as the case study of this work. This parameter is crucial because risk management aims at minimizing this cost while maintaining a low risk level. Our approach is generic and can easily be applied to the other risk model parameters. The main contributions of this paper are: (a) the design of a self-configuration strategy for VoIP risk models, (b) its specification based on an econometric feedback mechanism, (c) its integration into our runtime risk model, (d) its implementation into our Asterisk-based prototype (e) its evaluation based a set of simulation results.

The paper is consequently organized as follows. Section 2 describes the key concepts of runtime risk management in VoIP networks. Section 3 describes our self-configuration strategy and the considered parameters. Section 4 details the econometric feedback mechanism supporting our strategy. Section 5 presents the integration of this mechanism into our proof-of-concept prototype. Section 6 evaluates the performance of our solution through a set of experimental results. Related work are described in Section 7, and Section 8 concludes the paper and points out future research efforts.

2 Runtime risk management

Risk management is typically defined as the management process which consists in assessing risks and treating them i.e. taking the steps required for minimizing them to an acceptable level in the infrastructure. When we analyze existing work in the area of VoIP networks, we can observe that the first phase is covered by approaches for assessing threats (such as honeypot architectures and intrusion detection systems based on signatures, or based on anomalies [**?**]), and also by approaches for assessing vulnerabilities (such as auditing/benchmarking tools [**?**]). The second phase is covered by different types of treatments, in order to eliminate risks (risk avoidance) by applying best practices, to reduce and mitigate them (risk optimization) by deploying protection and prevention systems [**?**], to ensure against them (risk transfert) by subscribing an insurance contract or to accept them (risk retention) [**?**].

Runtime risk management aims at applying a continuous control of the infrastructure exposure to threats through the activation or deactivation of safeguards (as an instantiation of [**?**]). Rheostat instantiates such a dynamic schema [**?**]. We have previously shown how it can be applied to VoIP infrastructures [**?**]. Let a be a security attack part of A (the set of attacks), Rheostat quantifies the risk level $\mathcal{R}$ based on three parameters $\mathcal{P}(a)$, $\mathcal{E}(a)$ and $\mathcal{C}(a)$, as defined in Equation 1. $\mathcal{P}(a)$ stands for the potentiality of the threat associated to the attack a. $\mathcal{E}(a)$ defines the exposure of the infrastructure, which depends on the vulnerabilities of the system with respect to this attack. Finally, $\mathcal{C}(a)$ stands for

the consequences of a successful attack on the infrastructure resources. This last parameter quantifies the degradation of the assets.

$$\mathcal{R} = \sum_{a \in A} \mathcal{P}(a) \times \mathcal{E}(a) \times \mathcal{C}(a) \tag{1}$$

Rheostat exploits two algorithms for controlling the exposure of the infrastructure. These algorithms aim at maintaining the risk level to an acceptable level (less than a threshold $\mathcal{R}_{threshold}$ while reducing the costs of safeguards (see Equation 2, with the i^{th} safeguard being noted sf_i).

$$minimize(\sum_{i} cost(sf_i)) \; and \; \mathcal{R} < \mathcal{R}_{threshold} \tag{2}$$

The risk restriction algorithm activates security safeguards in order to reduce the risk level when the potentiality of a threat is high, while the risk relaxation algorithm deactivates theses safeguards when the potentiality is low in order to reduce the costs induced by safeguards.

Configuring the parameters of a risk model is an important and difficult activity, because the number of parameters may be high (parameters $\mathcal{P}(a)$, $\mathcal{E}(a)$ and $\mathcal{C}(a)$ are themselves dependent on other parameters) and also because parameters may vary with respect to the context. Several parameters are particularly hard to configure in our scenario. A first one is the impact of a security safeguard. It quantifies the capability of a security safeguard to protect the VoIP infrastructure with respect to a security threat. While this quantification is sometimes obvious (application of a specific patch), in most cases this parameter is difficult to quantify, in particular in case of unwanted communications (such as SPIT). We only focus here on the impact on the attack, not the impact on the threat itself. It is also interesting to quantify this second parameter, even if this impact is often low in case of attacks generated by bots. The second one is the cost of a security safeguard, which specifies how the safeguard deteriorates the performance of the service telephony. It is often defined in terms of service availability or usability. It is also possible to quantify it based on the number of innocent suspects that have to pass the security safeguard. At the extreme case, the cost of a safeguard can be considered as infinite when it consists in stopping the telephony service. This safeguard should only be executed when no alternative has been found for treating the considered risk. The last one is consequence of a successful attack, which is typically quantified in terms of confidentiality, integrity and availability. The objective is to determine if an attack will generate important damages or not on the VoIP infrastructure. While availability can be calculated in a dynamic manner in specific scenarios, privacy and integrity are more challenging and often require to be estimated by experts.

3 Self-configuration strategy

We propose in this paper to define a self-configuration strategy for improving runtime risk management in VoIP infrastructures (see Figure 1). Risk models

suffer from their complexity due to the high number of parameters. This automation is a key requirement in order to simplify this task and in order to adapt and refine risk model parameters with respect to their context. We consider an economic feedback mechanism to support our self-configuration schema. The objective is to take into account the experience in order to adapt the parameterization and to build a higher added-value modeling. As depicted in Figure 1, our

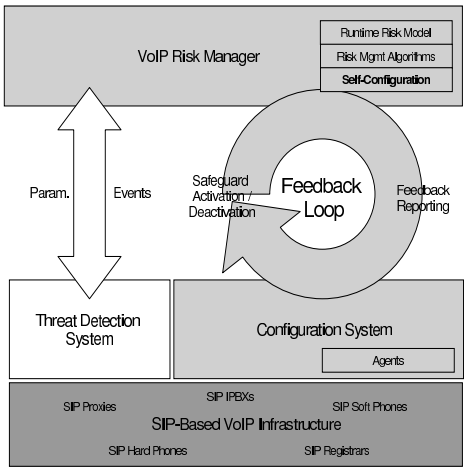


Fig. 1. Self-configuration strategy for VoIP runtime risk management

VoIP architecture is composed of three components: (1) a detection system responsible for quantifying the potentiality of security threats, (2) a risk manager responsible for selecting safeguards based on the runtime risk model and based on the management algorithms, and (3) a configuration system which executes the safeguards on the VoIP infrastructure. The self-configuration strategy permits to establish a feedback loop based on the reporting performed by agents deployed on VoIP equipments. Thanks to this reporting integrated into the risk model, each application of security safeguards permits to perform additional observations and to leverage our risk management strategy. We have voluntarily focused our study on the cost of safeguards, in particular on safeguards based on audio captcha tests [**?**]. It has been shown that several hundred million captchas are filled out every day, and that these captchas could represent a cost of one billion dollars in terms of productivity loss [**?**]. Even if this value is probably over-estimated, it illustrates the importance of well-configured risk models. After the application of a safeguard, the agent estimates its cost and reports it to the configuration server. The server collects and aggregates these statistics that are forwarded to the risk manager. The risk manager then exploits these data to refine the cost of the considered safeguard.

We consider a VoIP infrastructure based on the SIP protocol. SIP is an open standardized protocol for managing sessions in VoIP telephony [**?**]. It handles

the authentication and location of multiple participants, and supports the negotiation of media types using SDP (Session Description Protocol) messages. Let

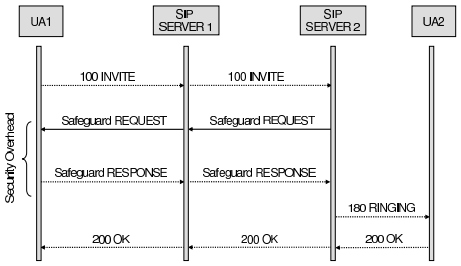


Fig. 2. SIP session initiation with security safeguard

consider the case of an agent UA1 establishing a VoIP communication with another agent UA2, as described in figure 2. The agent UA1 first sends an INVITE message to initiate a session with the second agent. If the potentiality of an attack is high, the SIP proxy of the second domain may require the application of a security safeguard in order to protect the VoIP equipments against attacks that could be generated by the first agent The agent UA1 is then invited to respond to this safeguard. For instance, in our case, the SIP proxy can apply an audio captcha test, such as requesting the typing of a specific code If the second agent UA2 provides a correct answer , the session initiation can continue normally with a RINGING message and an OK message.The application of the safeguard has introduced an additional overhead during the session initiation. In our scenario, we quantify the cost of the captcha safeguard in terms of delay.

4 Econometric feedback mechanism

Our self-configuration strategy is supported by an econometric feedback mechanism. The objective is to exploit the results of previous safeguard applications in order to refine the risk management model and determine the cost of next applications in a more efficient manner. A large variety of methods and techniques are available for performing such a forecasting with different performances, in particular in the area of econometry. We considered the commonly used ARMA analysis technique. While it presents some limitations, this technique is fully adequate with our runtime constraints, and our observations can easily be mapped to time series [**?**].

4.1 Refinement modeling

An ARMA model is typically defined as the combination of two models: the first one is an autoregressive model of order p and the second one is a moving

average model of order q. It can therefore be mathematically defined as given by Equation 4.

$$y_t = \sum_{i=0}^{p} \phi_i y_{t-i} - \sum_{j=0}^{q} \theta_j \epsilon_{t-j} + \epsilon_t \tag{3}$$

In this equation, the variable y_t stands for the forecasted value, while the $\{y_{t-i}\}$ variables represent the previous forecasted values. ϵ_t provides the error of the prediction method following the law $BB(0, \sigma_t)$. The $\{\phi_i\}$ and $\{\theta_i\}$ variables are the coefficients (positive or negative) to be determined. These coefficients can be estimated with the maximum likelihood method.

We apply the ARMA analysis technique in order to refine the cost of the security safeguards. We note $fcCost$ as the forecasted cost and $efCost$ as the effective cost of the security safeguard. In that case, the forecasted cost $fcCost_t$ at an instant time t is given by Equation 4 with ϵ_{t-j} standing for the difference between the effective safeguard cost $fcCost_{t-j}$ and the forecasted safeguard cost $efCost_{t-j}$ at time t and ϵ_{t-j} standing for the cost error.

$$fcCost_t = \sum_{i=0}^{p} \phi_i fcCost_{t-i} - \sum_{j=0}^{q} \theta_j \epsilon_{t-j} + \epsilon_t \tag{4}$$

As a consequence, the management algorithms (risk restriction algorithm and risk relaxation algorithm) permit to minimize the refined values corresponding the cost of security safeguards, while maintaining the risk level to an acceptable value, as described by Equation 2.

4.2 Analysis and validation

This analysis technique is typically specified into five phases, and includes a validation test [**?**]. We briefly describe below its application in our scenario of runtime risk management. The first phase consists in identifying and filtering periodicity; this task can be performed by analyzing simple and partial correlograms or by applying a dedicated test such as the augmented Dickey-Fuller test or the Philips-Perron test. The second phase permits to determine the orders p and q of the ARMA model ; this task is typically done again based on the analysis of simple and partial correlograms. The autocorrelation function measures the correlation between $efCost_t$ and $efCost_{t-k}$, and the influence of the other variables $(efCost_{t-i})_{0<i<k}$ having been withdrawn. The autocorrelation coefficient of order k is given by Equation 5.

$$\rho_k = \frac{cov(efCost_t, efCost_{t-k})}{\sigma_{efCost_t} \sigma_{efCost_{t-k}}} \tag{5}$$

The third phase estimates the coefficients $\{\phi_i\}$ and $\{\theta_i\}$ of the ARMA model. The coefficients weight respectively the variables $fcCost_{t-i}$ and ϵ_{t-j} (see Equation 4). Their estimation is obtained by exploiting the maximum likelihood method. The fourth phase represents the validation of the ARMA method. This

first consists in analyzing the coefficients and the residuals and then to apply the autocorrelation test of Box and Pierce using a static quantity Q which is given by this equation: $Q = n \sum_{k=0}^{K} \rho_k^2$. In this equation, n stands for the number of observations and ρ_k represents the autocorrelation coefficient of order k of the estimated residuals. This validation permits to determine the error term with respect to the sample size.The last phase consists in quantifying the predicted cost of the safeguard based on the established modelling, using Equation 4.

5 Prototype integration

We have integrated our self-configuration strategy into an Asterisk-based VoIP environment. We have exploited built-in Asterisk drivers, and implemented the self-configuration module based on AGI (Asterisk Gateway Interface) scripts using the AGI python toolkit. This prototype detects suspicious actors based on an anomaly detection algorithm detailed in [**?**]. This algorithm (monitoring package) identifies the presence of SPIT or other abnormalities based on Call Detail Records (CDRs). The identity of the suspicious actors is represented by the user account for a registered user and by the IP address for external calls. The monitoring package forwards the results to the risk management module. This one stores and manages the list of suspicious actors and assigns safeguards for each actor based on our runtime risk model. The AGI script takes the Aster-

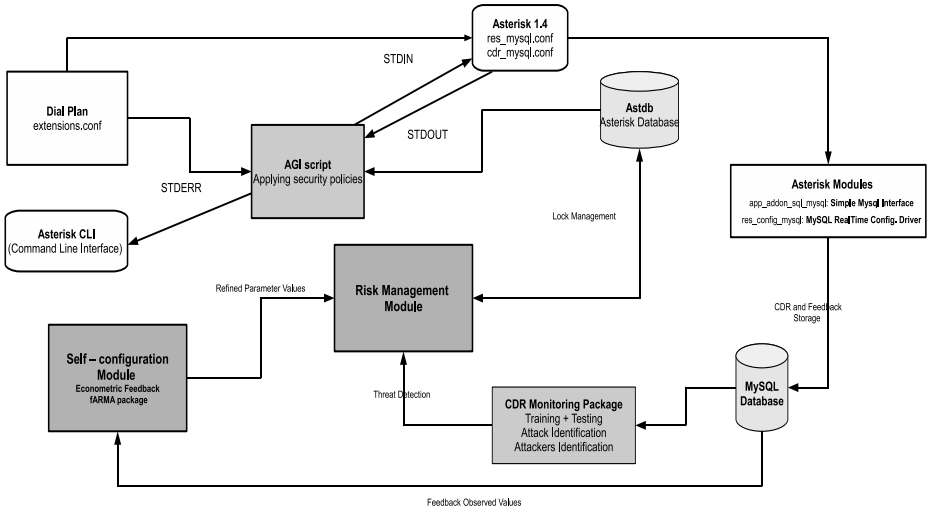


Fig. 3. Integration of the self-configuration strategy into our Asterisk-based prototype

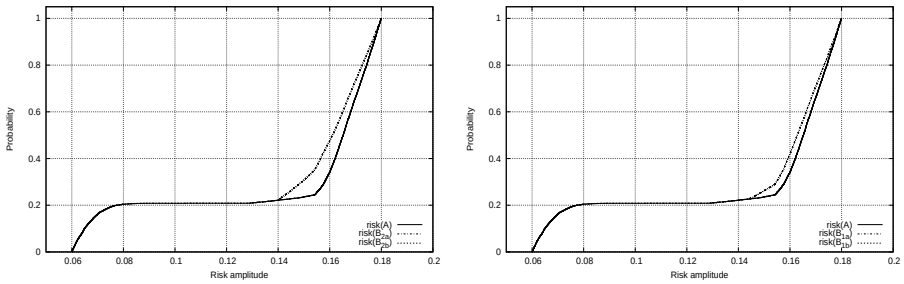
isk channel parameters as arguments and determines if any safeguard has to be applied before calling the extension. Our prototype currently supports several

safeguards, such as responding with a busy message for the first call tentative, asking to dial a specific DTMF tone in order to establish the call, and redirecting the call to another destination. After each application of a safeguard, the prototype stores the effective cost of the activated safeguards into the database. The self-configuration module analyses the series of cost values, and quantifies a refined value for the safeguard by applying our econometric feedback mechanism. It directly calls the ARMA (p,q) functions of the fArma package [**?**] and forwards the refined value to the risk management module. We have experimented this feedback mechanism with the audio captcha safeguard (5 x 30 samples) and have determined an error term which serves as a basis for simulation experiments.

6 Experimental results

We have considered the scenario of SPIT attacks, as SPIT is a very common threat in VoIP infrastructures. Our purpose is to evaluate the impact of our econometric feedback mechanism on the runtime risk management schema. The call arrival is represented by a Poisson law and a mean of 100 calls per unit of time. The call duration is represented by an exponential law and a mean of 10 seconds. The attacks are represented by 4 different types with increasing SPIT intensity (from 10 to 1000 SPIT calls per unit of time). We define from 5 to 20 different safeguards where each safeguard is characterized by three variables: the cost (representing the additional delay introduced by the safeguard, with an error term between 1% and 10%), the probability that a malicious call bypasses the safeguard (following a uniform distribution in the $[0.8; 1]$ interval), and the probability that an honest call bypasses the safeguard (following a uniform distribution in intervals between $[0.8; 1]$ (best cases) and $[0; 0.2]$ (worst cases). We have conducted 10,000 Monte Carlo simulations per scenario, which permits to reduce sufficiently the simulator error term. We use the same seed number for the pseudo-random number generation of all scenarios. Next, we expose a subset of our experimental results. We are in particular interested in evaluating the benefits and limits of the econometric feedback mechanism our runtime risk management performance. In a first series of experiments, we have investigated the impact of the feedback mechanism on the risk amplitude. Figure 4 represents the risk distribution for three different cases: a cost with an error term of 0% (scenario A), a cost with an error term of less than or equal to 5% (scenarios B_{1a} and B_{2a}), and a cost with an error term between 5% and 10% (scenarios B_{1b} and B_{2b}). We can clearly observe on the first subfigure 4(a) corresponding to an error term with a positive direction that the risk amplitude is higher with the scenario A than the two other scenarios B_{2a} and B_{2b}. The distinction between scenario A and the other scenarios starts with a risk amplitude greater than 0.14. The two curves corresponding to the scenarios B_{2a} and B_{2b} are converging to the same distribution. We observe the same phenomenon with the second subfigure 4(b). We have plotted the three same scenarios A, B_{1a} and B_{1b}, but in that case with a negative direction. The risk amplitude is once again higher with scenario A than with the two other scenarios B_{1a} and B_{1b}, and the distribution of these two last scenarios are also converging. However, the difference between scenario

A and the scenarios B_{1a} and B_{1b} is less important than in the first subfigure. These results are in coherence with our runtime risk management strategy. The objective of the risk management algorithms is to minimize the cost of activated security safeguards while maintaining the risk amplitude less than a threshold value, which permits to explain the experimental results observed in the two subfigures 4(a) and 4(b). In the first subfigure 4(a), the difference between scenario A and scenarios B_2 (B_{2a} and B_{2b}) is due to the activation of a security safeguard with an impact higher than effectively required. The error rate with a positive direction contributes to the selection of such a safeguard at an earlier stage, as its cost looks less expensive than the effective cost. In that case, the error term



(a) Error term with positive direction (b) Error term with negative direction

Fig. 4. Impact of feedback on risk amplitude

leads the runtime risk model to generate a restriction on the infrastructure exposure more important than required. This minimizes the risk amplitude in a more significant manner, but the cost due to activated safeguards is not optimized. This means the risk management module will activate security safeguards that are not necessarily required for protecting the VoIP infrastructure, and will introduce an additional delay in the service functioning. The difference between the B_{2a} and B_{2b} error terms has not been sufficient to modify the selection of security safeguards in these two scenarios. We can observe a similar behavior with the second subfigure 4(b), while we expected the opposite phenomenon: a risk amplitude less important with scenario A than with scenarios B_1. In this case, the cost of the security safeguard is decreased (negative direction) of up to 10%. The considered safeguard seems less expensive than its effective cost, this leads once again the risk management algorithms to select a more impacting safeguard than effectively required with respect to the potentiality of the threat. A lower risk amplitude with the two scenarios B_{1a} and B_{1b} does not mean the performance results are better, but that the risk management solution has underestimated the cost of the safeguard, which may generate a significant impact on the service performance. Another interesting question is to determine to what extent the econometric feedback mechanism impacts on the service performance. We have therefore evaluated in a second series of experiments, both the service availability and the total cost of security safeguards. We have plotted on figure 5

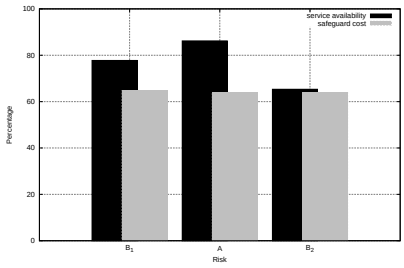


Fig. 5. Impact of feedback on service availability

a diagram representing these two metrics in a normalized manner (availability values and cost values estimated between 0% and 100%), for the three previously mentioned scenarios A, B_1 and B_2. We observe on this diagram that scenario A offers a lower effective cost due to security safeguards in comparison to scenarios B_1 and B_2. It also shows the best service performance with a value of up to 86%, while scenarios B_1 and B_2 provide respectively a value of up to 77% and up to 65%. Indeed, the VoIP infrastructure is overprotected in these two last scenarios because risk model parameters are not properly configured, which argues in favour of our refinement mechanism. It is also important to evaluate how the number of available security safeguards may impact on the service performance when the economic feedback mechanism is activated. We have quantified the service performance while varying the number of safeguards from 5 to 20. The risk management system behavior depends on the distribution of cost values on the set of security safeguards. The more the costs of two consecutive safeguards is important, the more the runtime risk management is sensitive to the error term. If we consider a distribution of costs sufficiently homogeneous amongst security safeguards, then a high number of security safeguards reduces the cost difference between two safeguards. As a consequence, the runtime risk model less tolerate on average the error term in that case. In the same manner, a low number of security safeguards increases the interval between the costs of two safeguards, and then reduces on average the sensitivity with respect to the error term value.

7 Related work

A few work really address risk management and its runtime instantiation in the area of VoIP infrastructures. Related work mentioned in section 2 only cover the risk management process in a partial manner, and do not integrate any risk model. This can be explained by the complexity to establish and configure risk models. Risk management is however a key requirement for protecting efficiently such a critical service. We have proposed in [**?**] a strategy capable to identify and treat risks at runtime in a VoIP environment. This solution is based on the extension of the Rheostat risk modeling and permits to prevent SPIT attacks based on a set of safeguards. We have observed in that context that the parameterization and maintenance of a risk model is expensive. We have therefore design

our self-configuration strategy in order to address this issue. The management of unwanted communications, in particular SPIT, has been extensively studied because of its importance for the future of VoIP. Quittek et. al. [**?**] apply hidden Turing tests on the caller side and compare their results to typical human communication patterns. For passing these tests, significant resource consumptions at the SPIT generating side would be required which contradicts the spammer's objective of placing as many SPIT calls as possible. VoIP SEAL [**?**] implements a two-stage decision process: the first stage contains modules which analyze a call only by looking at information which is available before actually answering the call. The second stage consists of modules which actually interacts with the caller or the callee to refine the detection. Since the second stage modules introduce some inconvenience, a scoring system is deployed at the first stage to determine if they will be used or not. Rather than Turing tests, other modules include white/black list, simultaneous calls, call rate, and URI's IP/domain correlation. Finally, the end-user feedback is taken into account if the SIP-client is instrumented for that. This work is the most similar to our work but does not explicitly propose a risk model. The end-user feedback could be easily integrated into our self-configuration schema. A survey of protection techniques against SPIT is given in [**?**]. The authors argue in favor of combining complementary techniques, which is fully in coherence with our dynamic solution and its automation. More elaborated econometric techniques [**?**] could be considered to instantiate our self-configuration approach, in particular techniques such as FFNN (Feed Forward Neural Network) and SVR (Support Vector Regression) [**?**].

8 Conclusions and Future Work

VoIP networks are exposed to multiple security threats, as they are less confined than traditional PSTN networks. Protection mechanisms are available, but their activation in such a critical environment may induce a signification deterioration of the service performance. Applying risk management in VoIP infrastructures is therefore a key requirement for protecting VoIP communications while maintaining the service usability. In that context, we have previously shown how the Rheostat runtime risk model can be extended to support risk management in VoIP infrastructures. We have also shown how the parameterization of such risk models can be difficult to maintain. In order to address this issue, we propose in this paper a self-configuration strategy for supporting risk management in VoIP networks. The objective is to dynamically adapt and refine the risk model parameters based on a econometric feedback mechanism. We have first remained the challenges of runtime risk management for such critical environments. We have then describe our self-management schema and shown how it can be deployed into a SIP-based architecture. We have mathematically detailed the econometric feedback mechanism. We have evaluated to what extent our solution can integrated into our implementation prototype. We have evaluate the performance of our approach through a set of simulation experiments. In particular, we have quantified the impact of feedback on risk amplitude and service performance. The error term due to poorly configured models can limit the ben-

efits of risk management and induce an additional overhead. Our automation permits to reduce the complexity of risk model parameterization and to perform a better treatment of risks in VoIP networks. As future work, we are interested in experimenting and evaluating alternative econometric techniques for adjusting our runtime risk models and improving the selection of security safeguards. We are also planning to investigate the deployment of our risk management approach into decentralized environments such as VoIP infrastructures exploiting the P2PSIP protocol.